

# **Exercises: Processing ChIP-Seq data**

## Licence

This manual is © 2018-19, Simon Andrews.

This manual is distributed under the creative commons Attribution-Non-Commercial-Share Alike 2.0 licence. This means that you are free:

- to copy, distribute, display, and perform the work
- to make derivative works

Under the following conditions:

- Attribution. You must give the original author credit.
- Non-Commercial. You may not use this work for commercial purposes.
- Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under a licence identical to this one.

Please note that:

- For any reuse or distribution, you must make clear to others the licence terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.
- Nothing in this license impairs or restricts the author's moral rights.

Full details of this licence can be found at

<http://creativecommons.org/licenses/by-nc-sa/2.0/uk/legalcode>

## Introduction

In this session we will go through how to process a small ChIP-Seq experiment. This will include:

- Quality control of raw sequence data
- Mapping to a reference genome with bowtie2
- Collation of raw and mapped QC with MultiQC

## Software

The software which will be used in this session is listed below. Software which requires a linux environment is indicated by an asterisk\*:

- Bowtie2\* (<http://bowtie-bio.sourceforge.net/bowtie2/>)
- Samtools (<http://www.htslib.org/>)
- FastQC (<http://www.bioinformatics.babraham.ac.uk/projects/fastqc/>)
- MultiQC (<http://multiqc.info/>)

## Data

The data in this practical comes from ENCODE accession ENCSR260AKX  
*C. elegans* genome data and GTF models come from Ensembl  
([https://www.ensembl.org/Caenorhabditis\\_elegans/](https://www.ensembl.org/Caenorhabditis_elegans/))

## Exercise 1: Quality Control

To start with we are going to use the FastQC program to look at the basic properties of the 3 fastq files we are going to process.

Move into the folder which contains the sequence data

```
cd ChIP-Data/Worm_ChIP_Mapping
```

Run fastqc on all of the files.

```
fastqc *fastq.gz
```

If you run `ls` you should be able to see several new HTML files generated by fastqc. You can open these in firefox to look at them.

```
firefox *.html &
```

(the `&` on the end puts the firefox program into the background, so it will keep running but allow you to enter more commands into the shell).

You can have a look through the QC results to try to answer the following questions:

- Did any of the QC modules trigger a warning or alert condition?
- Do the base call qualities provided by the sequencer suggest the data is high quality, or might it benefit from being quality trimmed?
- Are there any consistent sequence biases in the data?
- Is there any suggestion of the presence of adapter sequence which might need to be removed?
- Does the duplication level look as you'd expect for ChIP-Seq data?

## Exercise 2: Mapping and QC

We're going to manually go through the process of trimming and mapping one of our files. For a real analysis you'd repeat this for all of the files but to save time we're just doing a single file.

### Step 2.1 Creating a Genome Index

You will have been provided with a worm genome sequence in a file called `Caenorhabditis_elegans.WBcel235.fa`. Before you can run bowtie2 you will need to create an index file from this sequence which bowtie2 can then use.

```
bowtie2-build Caenorhabditis_elegans.WBcel235.fa celegans
```

The first option you supply here is just the name of the fasta file which contains your genome sequence. The second option is the base name for the index. This is the name you will need to supply when you run bowtie2, and you should also now be able to see the following files in your directory:

```
celegans.1.bt2          celegans.2.bt2          celegans.3.bt2
celegans.4.bt2          celegans.rev.1.bt2      celegans.rev.1.bt2
```

### Step 2.3 Adapter and Quality trimming

We are going to use the `trim_galore` trimming program to remove adapters and poor quality data. In some cases this will be more important than others. From looking at the FastQC reports you generated above, how important do you think this will be in this dataset?

We're going to trim and map one of the files. Here we'll use the `ENCFF394XPA_TBP1_Rep2.fastq.gz` file as an example, but you could pick any of them.

To trim the file we're using `trim_galore` so you just need to run:

```
trim_galore ENCFF394XPA_TBP1_Rep2.fastq.gz
```

This will generate a file called `ENCFF394XPA_TBP1_Rep2_trimmed.fq.gz` which is the trimmed version of the data and a trimming report which gives you the details of what was removed.

### Step 2.2 Running bowtie2

Now that we have the trimmed data we can send this to the bowtie2 mapping program to find the genomic location of each of the reads. For this we'll need the index files we made in step 2.1 and the trimmed file we made in step 2.3.

To do the mapping we will run (all on one line):

```
bowtie2 -p 2 -x celegans -U ENCFF394XPA_TBP1_Rep2_trimmed.fq.gz 2>trim.log |
samtools view -bS -o ENCFF394XPA_TBP1_Rep2.bam
```

The options in this command are as follows:

`-x` : Specifies the index file to use – this is the basename for the index which you created in step 2.1

`-p 2` : This tells bowtie2 that it can use 2 CPU cores on the machine. Short read mapping scales very efficiently across cores so if you have more cores available then you should use them to speed up your analysis.

`-U` : Specifies the fastq file of sequence reads to map. If you had paired end data and therefore 2 input files you would use `-1` and `-2` to specify the two files instead of `-U`

`| samtools view -Sb -o` : The final part of the command actually sends (pipes) the output of bowtie2 into another program called `samtools`. The purpose of this is to compress the raw sam format output from bowtie2 into the more compact bam format. The `-Sb` says that the input is SAM format and that that output should be BAM format. The `-o` is the name of the final output file.

The mapping will take a few minutes to complete. Once it's finished you should see the mapping statistics printed in the console, and you should see the bam file of aligned reads. You can see the set of files which have been generated by running:

```
ls -h
```

Have a look through and make sure you're happy that you know what each of the files is and which program produced them.

### Step 2.3 Running multiqc

To get a better overall picture of what's happened to all of our samples we can use `multiqc` to collate the QC and mapping statistics into a simple single report which we can view to assess any problems with the data. At the moment you have fastqc results for all 3 samples, and mapping and trimming statistics for just one. You can assemble these into a report by running

```
multiqc .
```

(note the dot at the end – this just says to process the files found in the current directory)

This should create a file called `multiqc_report.html` which you can view in firefox as before.

```
firefox multiqc_report.html &
```

Have a look through the combined statistics for the 3 samples and see if there are any of the metrics where they differ. To see a full report for all of the samples you can open the `multiqc_report.html` file which is the `BAM_Files` folder which has the details for all 3 samples.

See if you see anything which might be different between the different samples.

## Example Plots

So you know what you should be seeing here are copies of the plots you should generate in this practical:

### Step 2.3 Bowtie2 Mapping statistics

```
8140615 reads; of these:
  8140615 (100.00%) were unpaired; of these:
    43951 (0.54%) aligned 0 times
    6763351 (83.08%) aligned exactly 1 time
    1333313 (16.38%) aligned >1 times
99.46% overall alignment rate
```